The SCARE Frontier

Practical Side-Channel-Assisted Reverse-Engineering Attack on Protected AES Ciphers

G. Renault[1]

[1]School of Computer and Communication Sciences, EPFL, Switzerland

MSc Semester Project presentation, Jan. 2023

EPFL
LASEC

Table of Contents

Table of Contents

EPFL LASEC

What are Side-Channel Attacks (SCA) ?

Definition: Side-Channel Attack (SCA)

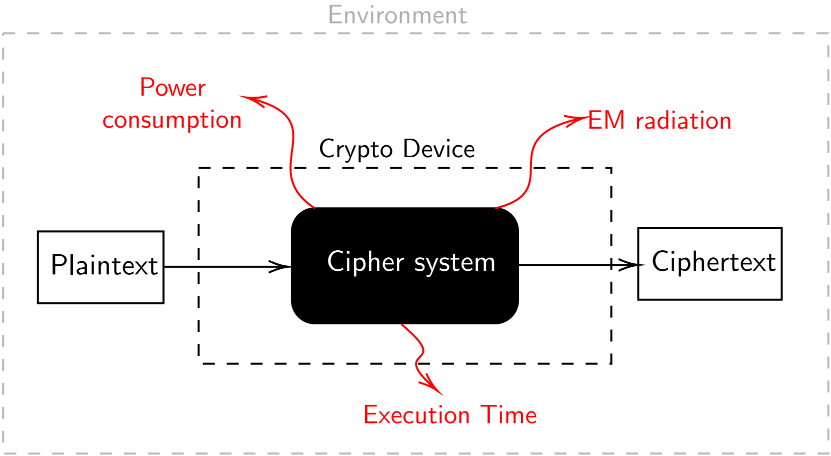
Attack exploiting information gained from the leakage of the cryptosystem's physical implementation.

- Introduced in scientific litterature by Kocher et al. [Koc96] in 1996.
- Possible side-channel leakages: Power consumption, EM radiation, Execution time, Sound, ...
- Used to extract secret information, such as encryption keys.
- Does not rely on vulnerabilities in the cryptographic algorithm itself.

EPFL LASEC

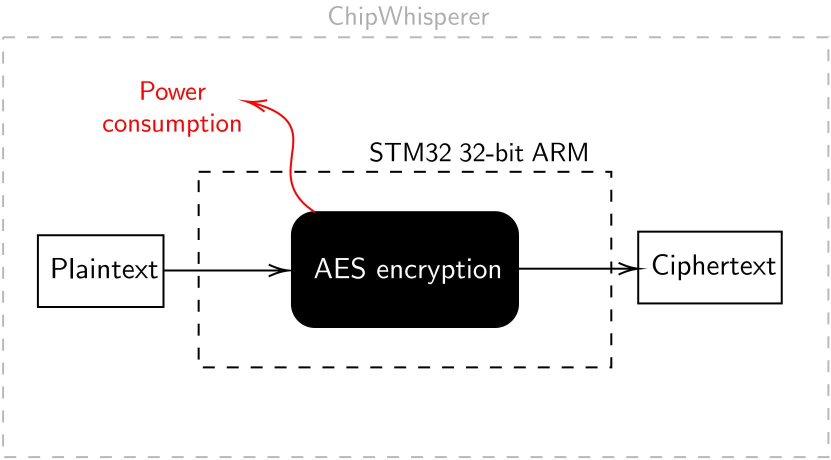
What are Side-Channel Attacks (SCA) ?

General Setup



What are Side-Channel Attacks (SCA) ?

Project Setup



Power (Consumption) Trace

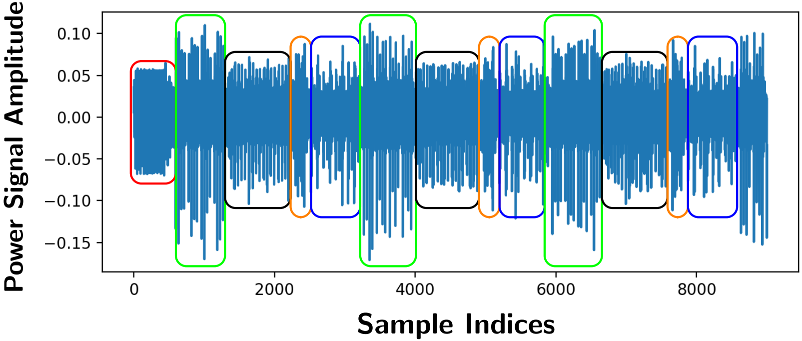
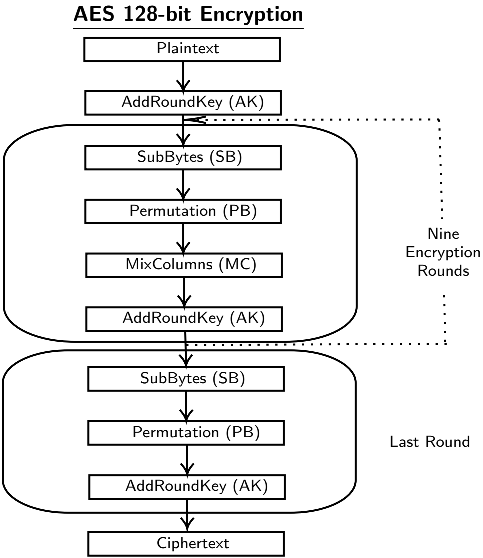


Figure: Average power trace of the first three rounds. In red, green, black, orange, blue the KS, AK, SB, PB, MC layers respectively. Trace corresponding to AK or SB layers are composed of 16 power spikes, each spike is associated with a byte operation.

Advanced Encryption Scheme (AES)



AES is considered to be secure but is not immune to SCA.

C implementation:

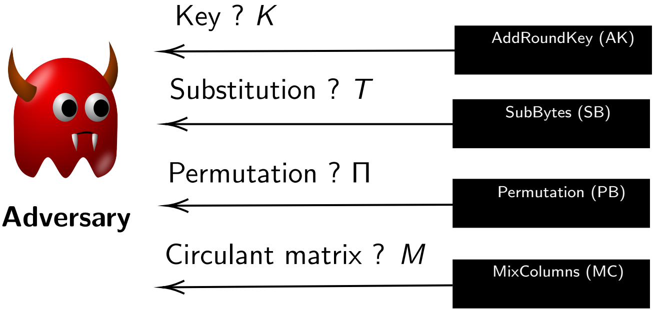
- TinyAES
- Byte-wise operation

Component functions are:

- Randomized
- Hidden

Adversary Model

Adversary's objective : Reverse-engineer the hidden AK, SB, PB, MC layers.



Some SCA to recover the secret keys:

- Differential Power Analysis (DPA) [KJJ99]: Compare the two power traces induced by the encryption of two different plaintexts. It is simple and inexpensive.
- Side-Channel Assisted Differential Plaintext Attack (SCADPA) [BJB18]: Chosen-plaintext DPA attack targeting the first round of the S-box layer.
- See-In-The-Middle (SITM) [BBH+19]: SCADPA extended to partially masked AES where the middle rounds remain unprotected.

SCARE attack to reverse-engineer the hidden component functions:

- [Nov03]: Reverse-engineer hidden structure of the A3/8 algorithm.
- [CIW13]: Complete SCARE of AES-Like Block Ciphers by Chosen Plaintext Collision Power Analysis.
- [RR13]: SCARE of Secret Ciphers with SPN Structures
- [CBB21]: Complete Practical Side-Channel-Assisted Reverse Engineering of AES-Like Ciphers.

Project Contributions

- Review SCARE paper from Caforio et al.
- New inexpensive and simple profiling technique.
- Apply SCARE attack on protected AES, implementing instructions shuffling at AK and SB layer using the previously mentioned profiling technique.

Table of Contents

SCARE Attack on AES

Back to implementation details

The attack targets the *AK* and *SB* layers.

```
; Round key addition
LD R1, [ADDR PT]
LD R2, [ADDR KEY]
XOR R1, R2
ST R1, [ADDR PT]
```

```
; Byte substitution
LD R1, [ADDR STATE]
ADD R2, R1, [ADDR SBOX]
LD R3, R2
ST R3, [ADDR STATE]
```

Figure: Assembly instructions of the *AK* and *SB* layers.

SCARE Attack on AES

Back to implementation details

The attack targets the *AK* and *SB* layers.

```
; Round key addition
LD R1, [ADDR PT]
LD R2, [ADDR KEY]
XOR R1, R2
ST R1, [ADDR PT]
```

```
; Byte substitution
LD R1, [ADDR STATE]
ADD R2, R1, [ADDR SBOX]
LD R3, R2
ST R3, [ADDR STATE]
```

Figure: Assembly instructions of the *AK* and *SB* layers.

By the Hamming-weight model [MOP07]:

- **AK**: $\rho(HW(R1), E(b_{i,AK})) > 0$

The attack targets the *AK* and *SB* layers.

```
; Round key addition
LD R1, [ADDR PT]
LD R2, [ADDR KEY]
XOR R1, R2
ST R1, [ADDR PT]
```

```
; Byte substitution
LD R1, [ADDR STATE]
ADD R2, R1, [ADDR SBOX]
LD R3, R2
ST R3, [ADDR STATE]
```

Figure: Assembly instructions of the *AK* and *SB* layers.

By the Hamming-weight model [MOP07]:

- **AK**: $\rho(HW(R1), E(b_{i,AK})) > 0$
- **SB**: $\rho(HW(R3), E(b_{i,SB})) > 0$

EPFL LASEC

SCARE Attack on AES

Back to implementation details

The attack targets the *AK* and *SB* layers.

```
; Round key addition
LD R1, [ADDR PT]
LD R2, [ADDR KEY]
XOR R1, R2
ST R1, [ADDR PT]
```

```
; Byte substitution
LD R1, [ADDR STATE]
ADD R2, R1, [ADDR SBOX]
LD R3, R2
ST R3, [ADDR STATE]
```

Figure: Assembly instructions of the *AK* and *SB* layers.

By the Hamming-weight model [MOP07]:

- **AK**: $\rho(HW(R1), E(b_{i,AK})) > 0$
- **SB**: $\rho(HW(R3), E(b_{i,SB})) > 0$

$\implies$ Create an algorithm computing/setting the Hamming weight of any state byte after the *AK* or *SB* layers.

SCARE Attack on AES

AK Attack

- **Recover** K: To recover the first key byte: iterate over the first plaintext byte and compute if its Hamming weight after AK layer is zero.

$$\begin{bmatrix} p_0 & p_4 & p_8 & p_{12} \\ p_1 & p_5 & p_9 & p_{13} \\ p_2 & p_6 & p_{10} & p_{14} \\ p_3 & p_7 & p_{11} & p_{15} \end{bmatrix} \xrightarrow{\text{AK}(0)} \begin{bmatrix} b_{0,AK(0)} & b_{4,AK(0)} & b_{8,AK(0)} & b_{12,AK(0)} \\ b_{1,AK(0)} & b_{5,AK(0)} & b_{9,AK(0)} & b_{13,AK(0)} \\ b_{2,AK(0)} & b_{6,AK(0)} & b_{10,AK(0)} & b_{14,AK(0)} \\ b_{3,AK(0)} & b_{7,AK(0)} & b_{11,AK(0)} & b_{15,AK(0)} \end{bmatrix}$$

- **Recover** K: To recover the first key byte: iterate over the first plaintext byte and compute if its Hamming weight after AK layer is zero.
- **Partial** Π **recovery**: DPA targeting the $AK(1)$ layer in order to find the positions of the plaintext byte activating the same column. We find Π up to row permutations.

$$\begin{bmatrix} p_0 & p_4 & p_8 & p_{12} \\ p_1 & p_5 & p_9 & p_{13} \\ p_2 & p_6 & p_{10} & p_{14} \\ p_3 & p_7 & p_{11} & p_{15} \end{bmatrix} \xrightarrow{\text{AK(0),..., AK(1)}} \begin{bmatrix} b_{0,AK(1)} & b_{4,AK(1)} & b_{8,AK(1)} & b_{12,AK(1)} \\ b_{1,AK(1)} & b_{5,AK(1)} & b_{9,AK(1)} & b_{13,AK(1)} \\ b_{2,AK(1)} & b_{6,AK(1)} & b_{10,AK(1)} & b_{14,AK(1)} \\ b_{3,AK(1)} & b_{7,AK(0)} & b_{11,AK(1)} & b_{15,AK(1)} \end{bmatrix}$$

- **Recover** K: To recover the first key byte: iterate over the first plaintext byte and compute if its Hamming weight after AK layer is zero.
- **Partial** Π **recovery**: DPA targeting the $AK(1)$ layer in order to find the positions of the plaintext byte activating the same column. We find Π up to row permutations.
- **Finding the candidates for** M: DPA targeting the $SB(2)$ layer and setting the state bytes after $SB(1)$ to some values (e.g. 255 or 0) in order to find a system of equations. Solving it gives $[a, b, c, d]$ up to its rotation.

- **Recover** K: To recover the first key byte: iterate over the first plaintext byte and compute if its Hamming weight after AK layer is zero.
- **Partial** Π **recovery**: DPA targeting the $AK(1)$ layer in order to find the positions of the plaintext byte activating the same column. We find Π up to row permutations.
- **Finding the candidates for** M: DPA targeting the $SB(2)$ layer and setting the state bytes after $SB(1)$ to some values (e.g. 255 or 0) in order to find a system of equations. Solving it gives $[a, b, c, d]$ up to its rotation.
- **Recover the substitution function** T: DPA finds some plaintext bytes satisfying the convergence property [BBH+19] by targeting the $SB(2)$ layer. From this, adversary can fill in the 256 entries of the substitution T.

Table of Contents

- **Masking**: protection mechanism that consist of performing operations with some random mask values at intermediate steps to obscure the input-output relationship.
- **Instructions shuffling**: protection mechanism that consists in randomly shuffling the executed instructions at some layers.

SCARE Attack on Protected AES

Instructions shuffling

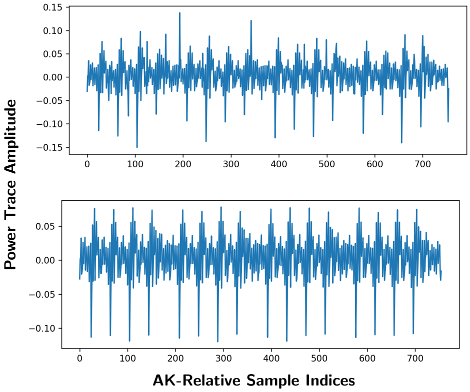


Figure: Top plot is one (unstable) power trace capture of the AK layer. Bottom plot is an average power trace of the AK layer. The spikes are similar as they are each an average spike induced over all the *AK* state bytes.

SCARE Attack on Protected AES
Profiling technique

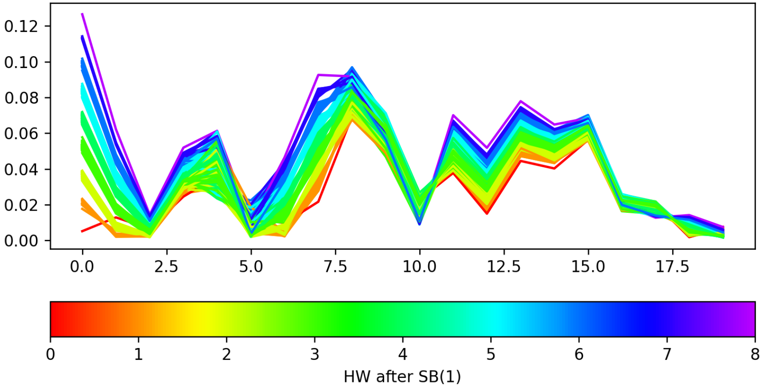


Figure: Power trace after the SB layer corresponding to encryption of the 0-th state byte.

- **Recover K**: To recover the key bytes **set**: iterate over all plaintext bytes, where all plaintext bytes are equal, and compute if one Hamming weight after AK layer is zero. We find K thanks to a set differentiation trick.

$$\begin{bmatrix} p_0 & p_4 & p_8 & p_{12} \\ p_1 & p_5 & p_9 & p_{13} \\ p_2 & p_6 & p_{10} & p_{14} \\ p_3 & p_7 & p_{11} & p_{15} \end{bmatrix} \xrightarrow{\text{AK}(0)} \begin{bmatrix} b_{0,AK(0)} & b_{4,AK(0)} & b_{8,AK(0)} & b_{12,AK(0)} \\ b_{1,AK(0)} & b_{5,AK(0)} & b_{9,AK(0)} & b_{13,AK(0)} \\ b_{2,AK(0)} & b_{6,AK(0)} & b_{10,AK(0)} & b_{14,AK(0)} \\ b_{3,AK(0)} & b_{7,AK(0)} & b_{11,AK(0)} & b_{15,AK(0)} \end{bmatrix}$$

SCARE Attack on Protected AES

- **Recover K**: To recover the key bytes **set**: iterate over all plaintext bytes, where all plaintext bytes are equal, and compute if one Hamming weight after AK layer is zero. We find K thanks to a set differentiation trick.
- **Partial Π recovery**: DPA targeting the $AK(1)$ layer in order to find the positions of the plaintext byte activating a same column. We find Π up to row-permutation and up to column-permutation

$$\begin{bmatrix} p_0 & p_4 & p_8 & p_{12} \\ p_1 & p_5 & p_9 & p_{13} \\ p_2 & p_6 & p_{10} & p_{14} \\ p_3 & p_7 & p_{11} & p_{15} \end{bmatrix} \xrightarrow{\text{AK(0),..., AK(1)}} \begin{bmatrix} b_{0,AK(1)} & b_{4,AK(1)} & b_{8,AK(1)} & b_{12,AK(1)} \\ b_{1,AK(1)} & b_{5,AK(1)} & b_{9,AK(1)} & b_{13,AK(1)} \\ b_{2,AK(1)} & b_{6,AK(1)} & b_{10,AK(1)} & b_{14,AK(1)} \\ b_{3,AK(1)} & b_{7,AK(0)} & b_{11,AK(1)} & b_{15,AK(1)} \end{bmatrix}$$

- **Recover K**: To recover the key bytes **set**: iterate over all plaintext bytes, where all plaintext bytes are equal, and compute if one Hamming weight after AK layer is zero. We find K thanks to a set differentiation trick.
- **Partial Π recovery**: DPA targeting the $AK(1)$ layer in order to find the positions of the plaintext byte activating a same column. We find Π up to row-permutation and up to column-permutation
- **Finding the candidates for M**: Similar as attack on non-protected AES but with the set differentiation trick.

SCARE Attack on Protected AES

AK, PB, MC, SB Attack(s)

- **Recover K**: To recover the key bytes **set**: iterate over all plaintext bytes, where all plaintext bytes are equal, and compute if one Hamming weight after AK layer is zero. We find K thanks to a set differentiation trick.
- **Partial Π recovery**: DPA targeting the $AK(1)$ layer in order to find the positions of the plaintext byte activating a same column. We find Π up to row-permutation and up to column-permutation
- **Finding the candidates for M**: Similar as attack on non-protected AES but with the set differentiation trick.
- **Recover the substitution function** T: Similar as attack on non-protected AES but with the set differentiation trick.

EPFL LASEC

Table of Contents

Conclusion

We first reviewed the practical SCARE paper from Caforio et al. After developing a new profiling technique, we extended this attack to protected AES where all layers could implement instructions shuffling.

Future work

- Extend to different implementations of AES.
- Extend to implementations of AES with a more complex hidden structure.
- Extend beyond AES ciphers.
- Apply ML methods to profiling and explore other leakages model.

Questions

Thank you for you attention.

References I

Shivam Bhasin, Jakub Breier, Xiaolu Hou, Dirmanto Jap, Romain Poussier, and Siang Meng Sim, *Sitm: See-in-the-middle side-channel assisted middle round differential cryptanalysis on spn block ciphers*, IACR Transactions on Cryptographic Hardware and Embedded Systems **2020** (2019), no. 1, 95–122.

Jakub Breier, Dirmanto Jap, and Shivam Bhasin, *Scadpa: Side-channel assisted differential-plaintext attack on bit permutation based ciphers*, 2018 Design, Automation & Test in Europe Conference & Exhibition (DATE), 2018, pp. 1129–1134.

Andrea Caforio, Fatih Balli, and Subhadeep Banik, *Complete practical side-channel-assisted reverse engineering of aes-like ciphers*, Cryptology ePrint Archive, Paper 2021/1252, 2021, `https://eprint.iacr.org/2021/1252`.

Christophe Clavier, Quentin Isorez, and Antoine Wurcker, *Complete scare of aes-like block ciphers by chosen plaintext collision power analysis*, Progress in Cryptology – INDOCRYPT 2013 (Cham) (Goutam Paul and Serge Vaudenay, eds.), Springer International Publishing, 2013, pp. 116–135.

Paul Kocher, Joshua Jaffe, and Benjamin Jun, *Differential power analysis*, Advances in Cryptology — CRYPTO' 99 (Berlin, Heidelberg) (Michael Wiener, ed.), Springer Berlin Heidelberg, 1999, pp. 388–397.

Paul C. Kocher, *Timing attacks on implementations of diffie-hellman, rsa, dss, and other systems*, Advances in Cryptology — CRYPTO '96 (Berlin, Heidelberg) (Neal Koblitz, ed.), Springer Berlin Heidelberg, 1996, pp. 104–113.

Stefan Mangard, Elisabeth Oswald, and Thomas Popp, *Power analysis attacks: Revealing the secrets of smart cards*, 01 2007.

R. Novak, *Side-channel attack on substitution blocks*, ACNS 2003. LNCS (J. Zhou, M. Yung, and Y. Han, eds.), vol. 2846, Springer, Heidelberg, 2003, p. 307–318.

Matthieu Rivain and Thomas Roche, *Scare of secret ciphers with spn structures*, Advances in Cryptology - ASIACRYPT 2013 (Berlin, Heidelberg) (Kazue Sako and Palash Sarkar, eds.), Springer Berlin Heidelberg, 2013, pp. 526–544.